1 The Gene Finding Program

1.1 Generalized Markov Model

The model used in gene finding is different from a pure Markov Model, but instead incorporates many "submodels" which can be of different types themselves, and then transitions between these submodels as in a normal Markov Model. In our case, the model used in gene finding consisted of a weight matrix model for sequence generation at the 3' end of the sequence, a decision tree model for sequence generation at the 5' end of the sequence, a 5th order Markov Model for the reading frame, and different models for the TATA boxes, poly-A signals and the other parts of the gene. Transitions with various probabilities were then used to join these various submodels together to get one overall Hidden Markov Model for sequence generation. In fact, this model consisted of "2 sides", for sequence generation, corresponding to the complimentary strands that would actually occur in the DNA. The submodels used all performed sequence generation with some probability based on the previous characters that had been generated. This would seem to indicate that the sequences generated would be "left to right" biased, but in fact, there is no more "left to right" bias than there is "right to left" bias. Only the way the model is described indicates any sort of bias. This model can be used to help in gene finding in two ways. It can be used to show which strings are generated with higher probability with respect to others. Also, Viterbi's algorithm can be used to find the most probable path through the states in the model. Once this has been observed, other sequences can be compared to the most probable sequence to decide if the sequence represents a certain gene or not.

1.2 Disadvantages of Hidden Markov Models

Consider the model shown in Figure ??.

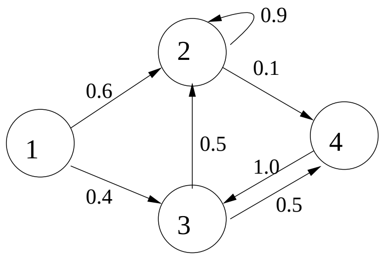

Figure 1: A simple Markov Model

Besides the states listed, each state may have various character emission probabilities associated with it, and a probability for the string terminating at that state. There will be some probability distribution on the set of strings which are generated by this model. Assuming that some string generated by the model reaches state 2, the expected length of the string from that point until it reaches state 4 is either 9 or 10. However, the most probable length of the string generated by the model is 1 (given that the model will go from state 2 to state 4, the most probable way for it to get there is to go directly from state 2 to state 4). The "loop" at state 2 does give some variability in the string lengths. In fact, the distribution of string lengths observed by having this sort of loop is a geometric distribution. This sort of distribution doesn't seem to mesh with the observed data in the distribution of intron/exon lengths within the human genome (or intron/exon lengths in other genomes for that matter). The intron (exon) lengths have a distribution with a "fatter than geometric" tail. Furthre, very short intron (exon) lengths seem to be even less probable than a geometric distribution would indicate. This problem can be overcome by incorporating enough states within the Markov Model to force minimal desired intron/exon lengths. The drawback to this is having a more complicated Markov Model/adding to the number of states to the Markov Model. Since the forward and backward algorithms described have the performace degraded by increasing the number of states, it would be desirable to try and have the number of states in the model minimized. Another drawback for the Markov Model is the difficulty it has in capturing the dependence the intron length has on the GC content of the sequence. While the exon length seems to independent of the GC content, introns are relatively long when there is low GC content, and relatively short when there is high GC content. This sort of phenomena is difficult to capture using only one Markov Model.

1.3 Parsing

Consider Figure **??** again. Let π be the initial state probability distribution, and let a_{ij} be the transition probabilities between the states. The output is a string, and each state is really a submodel of some type (perhaps Markov Model, Weight Matrix model, etc.). (Perhaps a promoter "state" is in fact a collection of states which will generate the promoter region). Given some $L > 0$ (a length):

1. Pick a start state q_i (with probability given by $\pi(i)$).
2. Pick a length d_i for the length of the substring output from this state, using the submodel to generate the string.
3. If the length of the substrings generated so far meets or exceeds L, stop.
4. Otherwise, pick a next state q_i' according to the probabilities a_{ij}, and goto (2).

The submodels generate subsequences with some certain length distributions, and usually, these distributions depend on which part of the sequence the submodel is generating (intron, exon, promoter, etc.).

Definition 1 *A* **Parse** ϕ *of a sequence* $S = S_1 \dots S_L$ *is a list of states* $q_1, \dots, q_k$ *and lengths* $d_1, \dots, d_k$ *(*d_i *the length of the string emitted by state* q_i*) such that* $\sum_{i=1}^{k} d_i \geq L$ *and* $\sum_{i=1}^{k-1} d_i < L$.

Then, given any parse ϕ, we can determine the probability of that parse, given the sequence S. It is the standard: $Pr(\phi|S) = \frac{Pr(\phi \wedge S)}{Pr(S)}$. We can then use a Viterbi-Like algorithm to find the highest probability parse ϕ given a sequence S, and then other forward/backward type algorithms to determine the probabilities that certain parts of the sequence are in the intron/exon/promoter/etc. phase. This method of finding the different functional regions within a string is surprisingly effective. It is worth noting that many of the choices within the framework of this generalized Markov Model are in fact deterministic and not probabilistic. This is because there are certain state transitions that we may always want to occur, based on the reading frame, or whether it was determined the region contained 1 or multiple exons, or other differences of this type.

1.4 More Problems with this Gene Finding Program

There are a number of problems with the gene finding program outlined here. There are some sequence phenomena that are difficult to identify, some that are difficult to model (as described before), and some errors that might arise that would cause the results to be questionable.

1. Short ORFs (open reading frames) are hard to recognize.
2. Sequencing errors lead to inaccurate prediction of the genes/ functional regions within the sequence.
3. Hard to identify UTRs and RNA genes.
4. The method is conservative, in the sense that the model will find and separate "highly probable"/"expected" phenomena, as opposed to finding something novel.
5. Usual overtraining of the model on the data could be a problem.

A couple of ideas that could be used in conjunction with this model might allow it to be used more successfully. The model could be used in conjunction with a protein database search: translate the DNA sequence into a protein sequence, and then attempt to match the resulting sequence with proteins in some available database. This method seems to work well in organisms which aren't "evolutionally far apart". Identification of the poly-A tail seems to help in recognizing the 3' end of the gene, but doesn't seem to consistently help in identifying the entire gene. The use of spliced alignments is another useful method in attempting to reconstruct genes.

2 Gene Finding using Comparative Genomics: PHRED and PHRAP

At this time, gene finding using Comparative Genomics is still in the early development stage, not well developed, and not well understood how this could be used in gene finding. There are a couple of tools available that would be helpful in at least the sequencing part of gene finding. PHRED is a tool which takes in a sequence (of undetermined makeup) and outputs an analog signal for each of the four possible nucleotides, representing their strength at different locations along the sequence. As an analog tool, there are a couple of shortfalls it has, namely deciding how long a "smear" of identical nucleotices is, and deciding which nucleotide would occur at some location if multiple nucleotides have moderate signals. Work has been done with assigning "quality scores" to various signals which can then be translated to a nucleotide sequence with some confidence. PHRAP is a tool which taks in a collection of sequences (partial substrings of some undetermind longer string, each possibly built using PHRED) and then attempts to combine them into some longer sequence. Again, the sequences should be chained together in a way so that the overlaps seem to agree "well", but deciding on some criteria to make this distinction is part of the difficulty.